

## Distributed Real Time Architecture for Data Placement in Wireless Sensor Networks

<sup>1</sup>Sanjeev Gupta and <sup>2</sup>Mayank Dave

<sup>1</sup>Doeacc Society, Autonomous body of DIT, Government of India, Chandigarh, India

<sup>2</sup>Department of Computer Engineering, National Institute of Technology, Kurukshetra, India

---

**Abstract: Problem statement:** Application scenarios of Wireless Sensor Networks (WSN) create interesting design challenges since the sensor nodes used to collect and communicate data may fail suddenly and unpredictably. The issue of real-time and reliable data delivery is extremely important for taking effective decisions in WSN. **Approach:** In this study we proposed architecture for reliable and real-time approach for data placement using sensor clusters. Instead of storing information in individual cluster heads as suggested in some protocols, in our architecture we suggest storing of information of all clusters within a cell in the corresponding base station. For data dissemination and action we have used Action and Relay Stations (ARS). **Results:** We developed programming model for formal specification of our architecture. **Conclusion:** Present model not only offers real time sensing and monitoring, but also provides real time dynamic decision making based on the sensed data. Moreover it prolongs network life time.

**Key words:** Modeling, SPIN, Promela, centralized clustering, base station, sink, cellular, formal verification

---

### INTRODUCTION

Reliable and correct behavior of sensor network based on infrastructure can be critical for human life, environment, decision-making and consequence of misbehavior can be catastrophic. Therefore development of reliable, stable and secure solutions that work provably correct even in unreliable and un-trusted environment is extremely important.

The analysis of the design should include verification, which provides the means to ensure the correctness of the design of such protocols. The protocol description under analysis should conform to its expected properties of reactivity and robustness. Verification process enables the designer to be confident that the formal description of the system does satisfy the system requirements of reactivity and robustness.

The enormous sets of conditions create a challenge for the development of systematic approaches to testing correctness or characterizing the performance of a protocol. Nevertheless, there is need for ensuring that the protocols deliver their design aim of robustness and reactivity and understanding of network protocols in general. We need techniques for systematic verification of protocol behavior, even in the face of the above challenges and obstacles.

The inherent complexity of concurrent real-time systems makes it necessary to employ mechanized, formally supported methods to analyze early life-cycle artifacts. In this context the main questions to be answered are whether the requirements are consistent and correct with the intended behavior of the system and whether the system's design correctly implements the requirements. It has been shown that state-based modeling and automatic model checking is an effective tool for answering these questions for concurrent reactive systems.

Given the increasing sophistication of Wireless Sensor network (WSN) algorithms and the difficulty of modifying an algorithm once the network is deployed; there is a clear need to use formal models to validate system performance and functionality prior to implementing such algorithms. The formalism should be intuitive and should support specifying the algorithm at an appropriate level of abstraction, so that a formal specification can be well understood and can provide a useful starting point for an implementation of the algorithm.

Our aim is to describe formally the architecture of reliable and real time data placement model and identify the performance criteria or correctness conditions that need to be investigated.

## MATERIALS AND METHODS

**Modeling and analysis of WSN:** Testing is used to check whether a given system realization conforms to an abstract specification. But it can be applied only after a prototype implementation of the system has been realized. Formal verification<sup>[1]</sup> as opposed to testing conducts an exhaustive exploration of all possible behaviors of the system. Formal verification works on models rather than implementations. Both techniques can be supported by tools. Model checking is an automated technique to validate correctness of a system by investigating a finite state model of the system. A model checker explores all states reachable from an initial state and validates a set of correctness properties on the model.

In model checking<sup>[2]</sup> algorithms executed by computer tools are used in order to verify the correctness of systems. Since sensor networks can be considered in terms of communicating finite state machines, they can be described as a set of concurrent communicating processes in Promela<sup>[3]</sup>. The Spin<sup>[4]</sup> is a model checking method and appropriate tool for verifying the sensor networks specifications. In this paper the model of reliable and real time data placement architecture has been formally described.

**Protocol Specification with Promela (process meta language):** Promela is a specification and verification modeling language. It provides a suitable specification language for concurrent systems. Concurrent software systems are erroneous and inherently complex to debug. Process interaction and process coordination are at the very basis of the language. It has three basic types of objects processes, variables and message channels.

Processes specify the behavior, variables define the environment in which the processes run and the message channels are used for modeling the inter-process communication. All communication between processes takes place via either message or shared variable.

**SPIN Model checker:** SPIN (Simple Model Interpreter) is a generic verification system that supports the design and verification of asynchronous process systems<sup>[4]</sup>. This model checker accepts design specifications written in Promela. It accepts correctness claims specified in the syntax of standard Linear Temporal Logic<sup>[5]</sup>.

**Visual interface to Promela-VIP:** The VIP<sup>[6]</sup> tool is a Java based graphical front end to the Promela

specification language and the SPIN model checker. VIP supports a visual formalism called v-Promela which extends the Promela language with a graphical notation to describe structural and behavioral aspects of a system.

**System architecture:** The proposed architectural framework<sup>[7]</sup> for data placement strategy is real time, reliable and distributed. It supports basic kind of data integrity and disaster management approaches for wireless sensor networks. Our architecture is portioned into modules each of which deals with various responsibilities of the overall system. The basic components and their functionalities used in our design are described briefly.

**Sensor nodes:** Sensors nodes are low-cost, low-power devices with limited sensing, computation and wireless communication capabilities. They can sense events in a circular coverage area with radius  $r_s$ . To save energy some sensors can be in sleeping state but they can be activated when it is necessary.

**Cluster structure:** One of crucial challenges in the organization of sensor networks is energy efficiency, because battery capacities of sensor nodes are severely limited and replacing the batteries is not practical. Clustering is a standard approach for achieving energy efficiency. It also reduces network contention by enabling locality of communication: nodes communicate their data over shorter distances to their respective cluster-heads. Not all nodes, but only the cluster-heads need to communicate far distances to their respective action and relay station ARS.

**Cell structure:** Within every cell there will be number of clusters. An equal-sized cluster is a desirable property because it enables an even distribution of control (e.g., data processing, aggregation, storage load) over cluster-heads; no cluster-head is over-burdened or under-utilized. Minimum overlap among clusters is desirable for energy efficiency because a node that participates in multiple clusters consumes more energy by having to transmit to multiple cluster-heads.

**Base station:** Our primary concern is the persistence of data and to minimize the amount of data lost due to failures of nodes as the emergency progresses (e.g., the fire spreads, rocks continue to bombard an area, or flood waters continue to grow). The base stations (BS) are responsible for data storage in a distributed real time database framework.

**Action and Relay Station (ARS):** The ARS are resource rich nodes equipped with better processing capabilities, higher transmission powers and longer battery life. The ARS nodes placed on the bordering areas of cells pass information received from various CHs to BS's of both the cells.

During disaster any ARS may be collapsed. But probability of collapsing all ARS's of cell is very small. Thus in case some BS's collapses there will be a little chance of the loss of invaluable information.

**Sink:** The sink acts as a bridge between wireless sensor network and physical world. The sink supervises and synchronizes the working of various components of WSN. It reports critical findings so that in case of some disaster or any other eventuality help from outside can reach the reported event area quickly. It monitors entire wireless network and depending upon feedback it sets the value of various controlling parameters like retention period so that the network can work efficiently.

**Modeling and specification of WSN:** Modeling and analysis of sensor networks require their formal specification<sup>[8]</sup>. In this work we have specified models of various components of our architecture in Promela. We make the following assumptions about wireless sensor nodes and wireless sensor network:

- Nodes are capable of measuring the signal strength of a received message<sup>[9]</sup>
- We assume that nodes have timers, but we do not require time synchronization across the nodes
- Nodes may fail-stop and crash, but we assume that the network does not get partitioned. New nodes can join the network
- A unique node is designated as a cluster-head of each cluster. Every node belongs to a cluster and no node belongs to multiple clusters
- Communication links are bi-directional and unreliable
- Underlying protocols ensure that nodes are aware of the set of nodes within their transmission radius.
- Network of Sensors must always be a connected graph
- There is (always) at least one communication way between any two sensors in the network
- The sensor network does not contain unreachable sensors
- Any sensor will eventually be connected to the rest of the network

**Model of sensor node:** Different actions of sensor nodes execute at specific timestamp when the sensor nodes are in active state. Periodically it will send the reading to its cluster head. Upon receiving a message the sensor node will update its local variables and may take on further actions like go into the sleep state or may change the periodicity of sensor readings. Other actions of the sensor node can be advertising itself as a new cluster head and some processing actions.

**Various possible states of sensor node (when working as normal node):**

- Probing (whenever sensor nodes wake up it will start probing neighborhood)
- Sleeping (to conserve energy of sensor node it will periodically go for sleep)
- Active (during this state it will probe environment for reading)

**Common Data objects of sensor node:**

- Sensor node location
- Energy the node has at the beginning
- Energy usage for packet transmission
- Energy usage for packet receipt
- Node status-active, sleeping
- Probing range of sensor node antenna's
- ARS location

**Data objects of sensor node working as CH:**

- Cluster head alive timer
- BatchTimer:
  - This timer will be relevant to the sensor node if it is currently the leader of its cluster
  - The CH will read all the records from its buffer after the expiry of this timer
  - It will then aggregate the reading information location wise
  - After aggregating the information CH will send the aggregated information to its respective ARS
- ProbeARSTimer:
  - The CH will send the probe message for the ARS to its environment
  - After expiry of this timer the cluster head will assume that there are no active ARS nodes
- EnergyLeftProbeTimer:
  - Periodically the cluster head will check amount of energy left
  - It will cease to be cluster head if energy left falls below a threshold limit



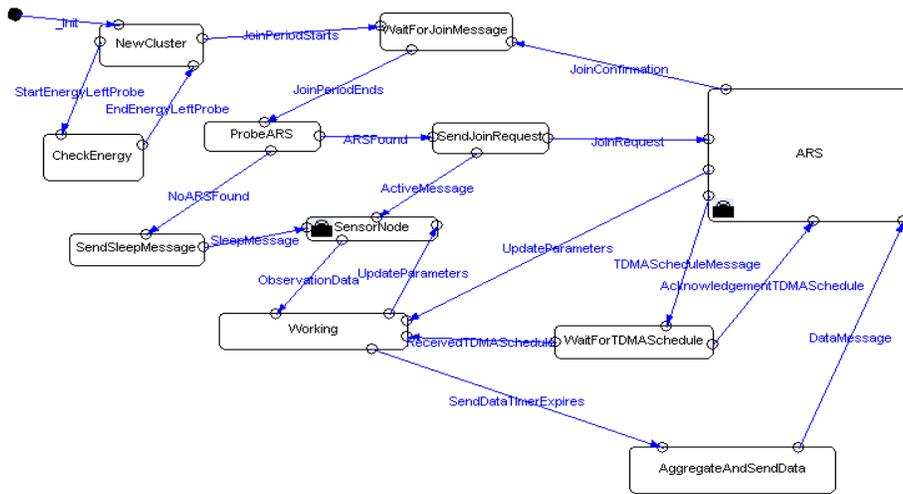


Fig. 3: Working of cluster class

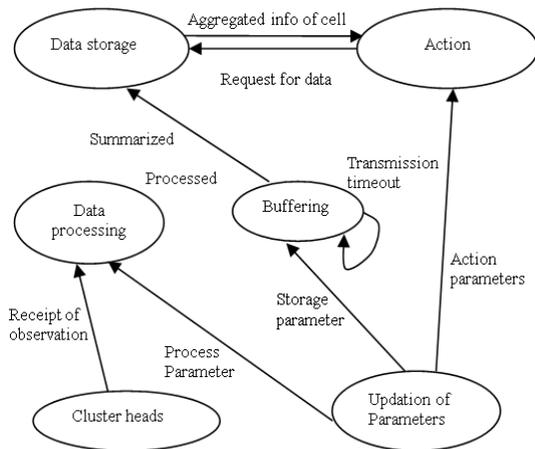


Fig. 4: Finite state machine

**Working of ARS:** As illustrated in Figure 4 the working of ARS will be:

- Maintaining a list of cluster (cluster heads) with whom it is communicating
- Relaying the information to adjacent base stations
- Periodically checking aliveness of the base stations
- Monitoring threshold value overshooting of which for a specified period is considered as emergency.

**Data objects of ARS class:**

- Status of ARS
- Action timer of the ARS node
- Radius of action cover area
- Status of base stations on each side of the edge

**Model of action unit class:** The action unit class consists of group of clusters and ARS. All the clusters belonging to action unit will send data directly to the ARS. The ARS will then forward the information received to the base station. In Fig. 5 working of action unit class is illustrated.

**Data objects of action unit class:**

- The list of cluster head locations
- Location Id of adjacent Base station
- List of neighboring ARS's within the probe area

**Model of base station:** The issue of energy saving, enhanced availability and reliability are extremely important for taking effective decision in a real world mission critical WSN based applications. Therefore the role of Base Station is very important in our model.

**Working of base station:**

- Periodically it will purge the old stored information
- Aggregating the information received over a period of time and storing the average readings only
- In case of emergency providing the information pertaining to the action area of the ARS to it
- Maintaining list of all ARS on the various edges of the cell of the base station

**Data objects of base station class:**

- Location of base station
- Database of reading records
- Purge timer on the firing of which it will delete old records from the database

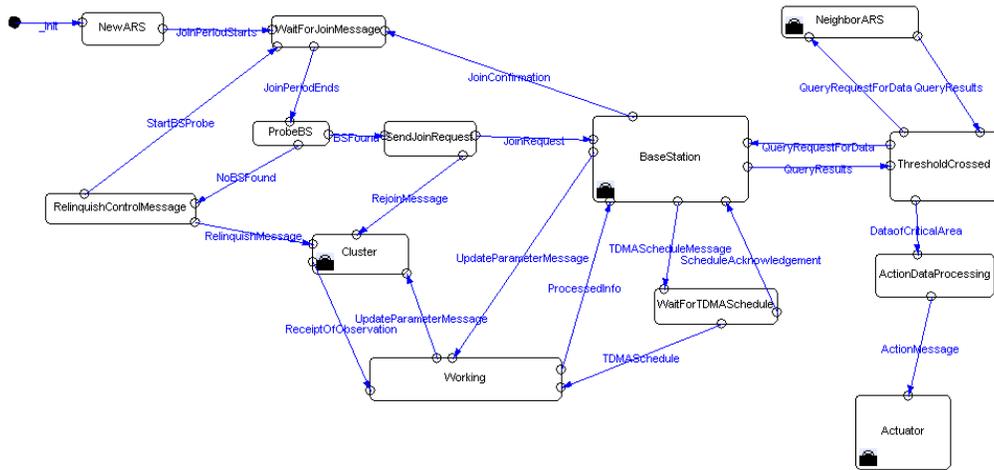


Fig. 5: Working of action unit class

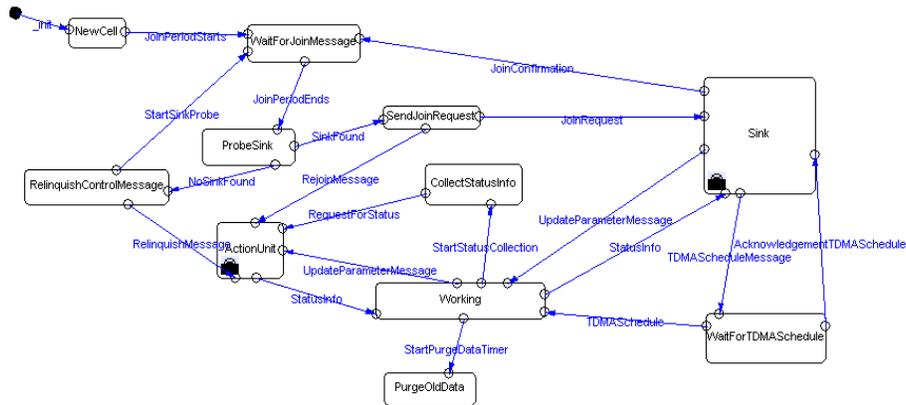


Fig. 6: Working of cell

**Model of cell:** A cell will have one base station and six Action Units. Figure 6 explains the working of cell.

**Data objects of cell class:**

- Location of base station
- List of locations of ARS nodes

**Model of sink:** The sink supervises the working of various components of WSN.

**Working of sink:**

- It will maintain a list of all base stations and their current status
- It will maintain a list of all ARS and their current status
- It will maintain a list of all clusters and their current status

- It will store statistical information to control the operation of wireless sensor network
- Periodically it will obtain the status of all the components of wireless sensor network
- On the basis of statistical information it will change the value of performance parameters to improve the behavior of wireless sensor network
- From time to time it will provide the current status of wireless sensor network to the controlling authority
- It will request the controlling authority to replace faulty components of WSN

**Data objects of sink class:**

- Location Id
- Storage of statistical information
- Initiates probe of wireless sensor network after expiry of enquiry status timer

**Model of RTWSN:** This class will represent entire WSN. This class will have one sink and many cells.

**Data objects of RTWSN class:**

- Location of sink
- List of locations of all cells
- List of all parameters

**Formal verification using SPIN:** We have used JSPIN<sup>[12]</sup> GUI interface for SPIN model checker for verifying our model because of its powerful model checking capabilities. All aspects of JSpin are configurable: at compile time, at initialization time through a configuration file or at runtime. The approach described in this work, has been to naively model the complete system and check it.

We have used state machine diagrams as a base for the PROMELA translation. Because the verification is bounded by the amount of physical memory available, we have reduced the complexity of the system. A large number of properties have been specified to safeguard the system described above. The modular approach allowed us to perform separate verifications for components of the model and compose them in order to prove properties for the whole model.

**RESULTS**

For even moderately-sized networks with tens of nodes, it is extremely difficult to analytically model the interactions between all the nodes. Therefore to show effectiveness of the proposed model the simulation software has been developed to conduct the said experiments. In the simulation 96 sensor nodes were randomly distributed in a cellular region of 120×120 m. To simplify the analysis we have assumed maximum of 24 clusters and maximum of 4 clusters per ARS. The data transmissions from sensor nodes were simulated until all sensor nodes died. For the experiments described here the parameters as described in Table 1 were used.

Table 1: Simulation parameters

| Energy assumptions                           | Value                                       |
|--|---|
| $d_0$ : Threshold transmission distance      | 80 m  |
| $E_{elec}$ : Electronics energy              | 50 nJ bit <sup>-1</sup>                     |
| $E_g$ : Amplifier energy factor (free space) | 100 pJ bit <sup>-1</sup> m <sup>-2</sup>    |
| $E_r$ : Amplifier energy factor (multipath)  | 0.0013 pJ bit <sup>-1</sup> m <sup>-4</sup> |
| $E_{da}$ : Aggregation energy                | 5 nJ bit <sup>-1</sup> signal <sup>-1</sup> |
| Initial battery of all the nodes             | 0.5 J                                       |
| CH proposal packet size                      | 800 bits                                    |
| CH Intimation packet size                    | 400 bits                                    |
| TDMA packet size                             | 800 bits                                    |
| Sensed message size                          | 600 bits                                    |

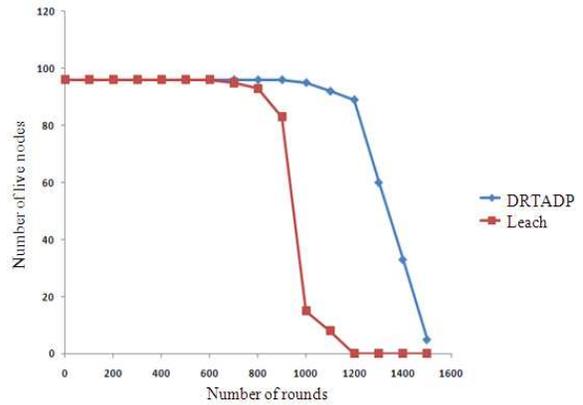


Fig. 7: Number of live nodes over no. of Rounds

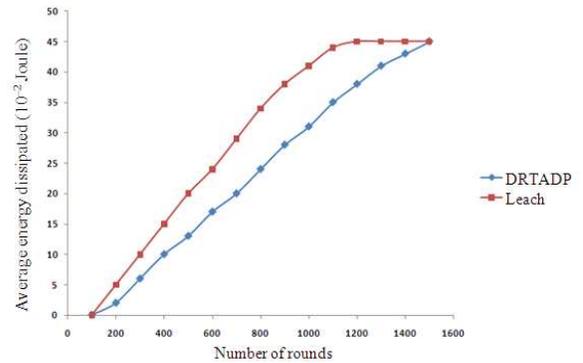


Fig. 8: Average energy dissipation over number of rounds

For energy dissipation the model explained in<sup>[13]</sup> have been used, that depends on the distance between the transmitter and receiver.

The performance of proposed model has been compared with popular Centralized-Clustering algorithm Leach-C taking exactly same environment and same assumptions.

**DISCUSSION**

Although Leach-C appears to be promising centralized clustering algorithm, there are some areas for making protocol more energy efficient. Instead of all the sensor nodes of the cell communicating to single BS, the proposed distributed model will be more energy efficient because average distance between cluster member nodes and respective ARSs will be much shorter than average distance between sensor nodes of clusters and the single base station of centralized clustering approaches.

Moreover instead of communicating and checking battery level of all the nodes of the cell to decide clustering, the ARS will only communicate with nodes reporting to it. Thus clustering process will be completed much faster and results in minimum use of invaluable energy of nodes and ARS.

The number of rounds that first node and last node dies as well as average energy dissipation is used as a key indicator to evaluate the proposed system. Experiment shows that our proposed model increases the lifetime of the network as illustrated in figures 7 and 8. The scalability is also very easy to achieve in our model.

### CONCLUSION

This study is about the application of PROMELA language and the SPIN tool to verify a practical implementation of our proposed model. It has been shown how a formal sensor behavior description can be transformed into Promela notation. The the sensor networks has been described as a set of concurrent communicating processes in Promela for further verification using SPIN. The SPIN has been applied for the analysis of our model to verify that it satisfies desired properties and/or is consistent with given global constrains.

We are also extending our simulation software to verify other aspects of our model like reliable, real time data retrieval and fault tolerance. Looking forward, one can expect that a lot more power-efficient designs will be produced from this framework. Larger tests are needed to determine the optimal transition period to the change the number of clusters in the network. Similarly there should be synchronization among different sensors reporting the sensed events to the ARS.

### REFERENCES

1. Haverkort, B.R., H. Hermans and J.P. Katoen, 2000. On the use of model checking techniques for quantitative dependability evaluation. Proceeding of the IEEE Symposium on Reliable Distributed Systems, Oct. 16-18, IEEE Computer Society, Washington DC., USA., pp: 228-238. <http://portal.acm.org/citation.cfm?id=831103>
2. Bérard, B., M. Bidoit, A. Finkel and F. Laroussinie, 1999. Systems and Software Verification: Model Checking Techniques and Tools. Springer-Verlag New York, Inc., ISBN: 3-540-41523-8.L, pp: 190.
3. Holzmann, G.J., 1990. Design and Validation of Computer Protocols. Prentice Hall, New Jersey, ISBN: 0-135-39925-4, pp: 512.
4. Holzmann, G.J., 2003. The SPIN model checker: Primer and Reference Manual. Addison Wesley. ISBN: 9780321228628, pp: 608.
5. Lamport, 1994. The temporal logic of actions. ACM Trans. Programm. Languages Syst., 16: 872-923.
6. Moataz Kamel and Stefan Leue, January 2000. VIP: A visual editor and compiler for v-Promela. Tools and Algorithms for the Construction and Analysis of Systems. Springer, Berlin/Heidelberg, ISBN: 978-3-540-67282-1, pp: 471-486.
7. Gupta, S. and M. Dave, 2008. Real time approach for data placement in wireless sensor networks. Int. J. Elect. Circ. Syst., 2: 132-139. <http://www.waset.org/journals/waset/v38/v38-59.pdf>
8. Mohamed G. Gouda and Young-Ri Choi, 2007. A state-based model of sensor protocols. Principles of Distributed Systems. Springer, Berlin/Heidelberg, ISBN: 978-3-540-36321-7.
9. Hill, J., R. Szewczyk , A. Woo, S. Hollar, D. Culler and K. Pister, 2000. System architecture directions for networked sensors. ACM SIGPLAN, 35: 93-104.
10. Heinzelman, W.B., A.P. Chandrakasan and H. Balakrishnan, 2002. An application-specific protocol architecture for wireless microsensor networks. IEEE Trans. Wireless Commun., 1: 660-670. DOI: 10.1109/TWC.2002.804190
11. Gamwarige, S. and E. Kulasekere, 2007. Optimization of cluster head rotation in energy constrained wireless sensor networks. Proceeding of the IFIP International Conference on Wireless and Optical Communications Networks, July 2-4, IEEE Xplore Press, Singapore, pp: 1-5. DOI: 10.1109/WOCN.2007.4284155
12. Moti Ben-Ari, 2009. Development Environments for Spin and Erigone. <http://stwww.weizmann.ac.il/g-cs/benari/jspin/>
13. Rappaport, T., 2001. Wireless Communications: Principles and Practice. 2nd Edn., Prentice-Hall, ISBN: 0130422320, pp: 736.